

**UNITED STATES PATENT APPLICATION**

of

**Christopher G. Kaler,**

**John Shewchuk,**

**and**

**Giovanni Della-Libera**

for

**FLEXIBLE ELECTRONIC MESSAGE SECURITY MECHANISM**

WORMHAIN, INTELLOUER & SELLER I  
A PROFESSIONAL CORPORATION  
ATTORNEYS AT LAW  
1000 EAGLE GATE TOWER  
60 EAST SOUTH TEMPLE  
SALT LAKE CITY, UTAH 84111



[0004] Currently, computing systems are so interconnected that one computing system is literally capable of communicating with any one of many millions of other computing systems spread throughout the globe. This is useful as we are now able to communicate more readily. However, this high level of interconnectivity also exposes us to security problems. For example, often it is necessary to verify that a computing device or associated user is truly the same entity that they purport to be in a process called authentication. Also, it is often important to validate the integrity of an electronic message to be sure that the electronic message has not been compromised during transmission.

[0005] Improvements in security mechanisms are of significant benefit since breaches in security can cause much harm, financial and otherwise, to entities who rightfully desire secure electronic communications. The principles of the present invention improve security over conventional security technologies as will be described in further detail below.

## **BRIEF SUMMARY OF THE INVENTION**

[0006] The principles of the present invention relate to mechanisms for providing reliable and flexible security mechanisms when communicating using electronic messages. The electronic message may have multiple different types of credentials. The electronic message may include the encoding format and type of each of the credentials thus allowing for convenient access of the credential by either the recipient computing system, or by an intermediary computing system. The electronic message may also include multiple signatures that were each signed using a different credential. The signatures may each have a reference to a location even external to the electronic message. The recipient computing system may evaluate the external credential against the signature to determine whether tampering of the electronic message may have occurred.

[0007] Additional features and advantages of the invention will be set forth in the description which follows, and in part will be obvious from the description, or may be learned by the practice of the invention. The features and advantages of the invention may be realized and obtained by means of the instruments and combinations particularly pointed out in the appended claims. These and other features of the present invention will become more fully apparent from the following description and appended claims, or may be learned by the practice of the invention as set forth hereinafter.

## **BRIEF DESCRIPTION OF THE DRAWINGS**

[0008] In order to describe the manner in which the above-recited and other advantages and features of the invention can be obtained, a more particular description of the invention briefly described above will be rendered by reference to specific embodiments thereof which are illustrated in the appended drawings. Understanding that these drawings depict only typical embodiments of the invention and are not therefore to be considered to be limiting of its scope, the invention will be described and explained with additional specificity and detail through the use of the accompanying drawings in which:

[0009] Figure 1 illustrates a suitable computing system in which the principles of the present invention may be employed;

[0010] Figure 2 illustrates a flowchart of a method for securely transmitting an electronic message in accordance with the principles of the present invention;

[0011] Figure 3 illustrates a data structure of an electronic message having multiple different types of credentials in the header of the electronic message in accordance with the principles of the present invention;

[0012] Figure 4A illustrates a network environment in which multiple different credentials are used to identify a source computing system to a particular recipient computing system in a model called herein the “multiple credential – single recipient model”;

[0013] Figure 4B illustrates a network environment in which different credentials in the electronic message may be used to identify the source computing system to an intermediary computing system and to identify the source computing device to a recipient computing system in a model called herein the “serial credential model”;

[0014] Figure 4C illustrates a network environment in which different credentials in the electronic message may be used to identify the source computing system to different recipient computing systems in a model called herein the “parallel credential model”;

[0015] Figure 4D illustrates a network environment which combines all of the models of Figures 4A, 4B and 4C; and

[0016] Figure 5 illustrates a credential semantic inheritance tree in accordance with the principles of the present invention.

## **DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS**

[0017] The principles of the present invention relate to methods, systems, computer program products, and data structures that allow more secure communications of an electronic message.

[0018] Multiple different credentials and/or signatures based on different credentials may be included in a header portion of a single electronic message. These different signatures and/or credentials may be used by different recipient computing systems, by a single recipient computing system, or even by different computing systems along a routing path of the electronic message.

[0019] The electronic message may include an identification of an encoding algorithm and the type of credential included in the electronic message. Accordingly, multiple different credentials may be included that have different encoding. The recipient computing system may decode and process the credential as appropriate given the identification of the encoding algorithm and the type of credential.

[0020] Also, the electronic message may include a pointer that references a credential that is accessible to the recipient computing system, either within the same electronic message, or from some other location. The recipient computing system may then compare the referenced credentials from the credentials used to generate the signature. If a match occurs, then the data signed can be associated with the credentials. Accordingly, the integrity of any statement made in the credentials such as identity, rights, and so forth, may be verified.

[0021] Embodiments within the scope of the present invention include computer-readable media for carrying or having computer-executable instructions or data structures

stored thereon. Such computer-readable media can be any available media which can be accessed by a general purpose or special purpose computer. By way of example, and not limitation, such computer-readable media can comprise physical computer-readable media such as RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to carry or store desired program code means in the form of computer-executable instructions or data structures and which can be accessed by a general purpose or special purpose computer. The computer-readable media may include at least partially run-time memory that holds data structures and modules that exist at run-time. The computer-readable media may also include persistent memory that stores data structures that persist regardless of the existence and/or reliability of externally supplied power.

**[0022]** When information is transferred or provided over a network or another communications connection (either hardwired, wireless, or a combination of hardwired or wireless) to a computer, the computer properly views the connection as a computer-readable medium. Thus, any such a connection is properly termed a computer-readable medium. Combinations of the above should also be included within the scope of computer-readable media. Computer-executable instructions comprise, for example, instructions and data which cause a general purpose computer, special purpose computer, or special purpose processing device to perform a certain function or group of functions.

**[0023]** Figure 1 and the following discussion are intended to provide a brief, general description of a suitable computing environment in which the invention may be implemented. Although not required, the invention will be described in the general context of computer-executable instructions, such as program modules, being executed by



computers in network environments. Generally, program modules include routines, programs, objects, components, data structures, and the like, that perform particular tasks or implement particular abstract data types.

**[0024]** Those skilled in the art will appreciate that the invention may be practiced in network computing environments with many types of computer system configurations, including personal computers, hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. The invention may also be practiced in distributed computing environments where tasks are performed by local and remote processing devices that are linked (either by hardwired links, wireless links, or by a combination of hardwired or wireless links) through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

**[0025]** With reference to Figure 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a conventional computer 120, including a processing unit 121, a system memory 122, and a system bus 123 that couples various system components including the system memory 122 to the processing unit 121. Throughout this description, element numbers begin with the same number as the figure in which the corresponding elements were first introduced. For example, all of the element numbers in Figure 1 are numbered in the 100's while the element numbers in Figure 2 are number in the 200's, and so forth.

**[0026]** The system bus 123 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read only memory (ROM) 124 and random

access memory (RAM) 125. A basic input/output system (BIOS) 126, containing the basic routines that help transfer information between elements within the computer 120, such as during start-up, may be stored in ROM 124.

[0027] The computer 120 may also include a magnetic hard disk drive 127 for reading from and writing to a magnetic hard disk 139, a magnetic disk drive 128 for reading from or writing to a removable magnetic disk 129, and an optical disk drive 130 for reading from or writing to removable optical disk 131 such as a CD-ROM or other optical media. The magnetic hard disk drive 127, magnetic disk drive 128, and optical disk drive 130 are connected to the system bus 123 by a hard disk drive interface 132, a magnetic disk drive-interface 133, and an optical drive interface 134, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer-executable instructions, data structures, program modules and other data for the computer 120. Although the exemplary environment described herein employs a magnetic hard disk 139, a removable magnetic disk 129 and a removable optical disk 131, other types of computer readable media for storing data can be used, including magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, RAMs, ROMs, and the like.

[0028] Program code means comprising one or more program modules may be stored on the hard disk 139, magnetic disk 129, optical disk 131, ROM 124 or RAM 125, including an operating system 135, one or more application programs 136, other program modules 137, and program data 138. A user may enter commands and information into the computer 120 through keyboard 140, pointing device 142, or other input devices (not shown), such as a microphone, joy stick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 121 through a serial port interface 46

coupled to system bus 123. Alternatively, the input devices may be connected by other interfaces, such as a parallel port, a game port or a universal serial bus (USB). A monitor 147 or another display device is also connected to system bus 123 via an interface, such as video adapter 148. In addition to the monitor, personal computers typically include other peripheral output devices (not shown), such as speakers and printers.

[0029] The computer 120 may operate in a networked environment using logical connections to one or more remote computers, such as remote computers 149a and 149b. Remote computers 149a and 149b may each be another personal computer, a server, a router, a network PC, a peer device or other common network node, and typically include many or all of the elements described above relative to the computer 120, although only memory storage devices 150a and 150b and their associated application programs 136a and 136b have been illustrated in Figure 1. The logical connections depicted in Figure 1 include a local area network (LAN) 151 and a wide area network (WAN) 152 that are presented here by way of example and not limitation. Such networking environments are commonplace in office-wide or enterprise-wide computer networks, intranets and the Internet.

[0030] When used in a LAN networking environment, the computer 120 is connected to the local network 151 through a network interface or adapter 153. When used in a WAN networking environment, the computer 120 may include a modem 154, a wireless link, or other means for establishing communications over the wide area network 152, such as the Internet. The modem 154, which may be internal or external, is connected to the system bus 123 via the serial port interface 146. In a networked environment, program modules depicted relative to the computer 120, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are

exemplary and other means of establishing communications over wide area network 152 may be used.

[0031] While Figure 1 illustrates an example of a computing system that may implement the principles of the present invention, any computing system may implement the features of the present invention. In the description and in the claims, a “computing system” is defined as any hardware component or components that are capable of using software to perform one or more functions. Examples of computing systems include desktop computers, laptop computers, Personal Digital Assistants (PDAs), telephones, or any other system or device that has processing capability.

[0032] Figure 2 illustrates a method 200 for performing electronic messaging in a secure manner. Some of the acts and the step of the method 200 are performed by a sender computing system that sends an electronic message. Those acts and that step are generally listed in the left column of Figure 2 under the heading “SENDER”. Other acts of the method 200 are performed by a receiver computing system that receives the electronic message. Those acts are generally listed in the right column of Figure 2 under the heading “RECEIVER”.

[0033] The method 200 includes a functional, result-oriented step for constructing an electronic message so as to provide increased security (step 210). This functional, result-oriented step may include any corresponding acts for accomplishing this result. However, in the illustrated embodiment, the step 210 includes corresponding acts 211 through 218. An example electronic message data structure is illustrated in Figure 3 as electronic message 300. The method of Figure 2 will be described with frequent reference to the electronic message data structure of Figure 3.

[0034] The method 200 includes an act of designating at least one destination address in the electronic message (act 211). The destination address corresponds to one or more recipient computing systems. Referring to Figure 3, the electronic message includes a header field 310 and a body field 330. The body field 330 may contain the content of the information desired to be communicated to the recipient(s), while the header field 310 contains information that facilitates proper and secure transport and processing of the electronic message. The header field 310 includes a destination address field 311 that contains the destination address or addresses of one or more desired recipients of the electronic message.

[0035] The header field 310 also includes an encryption manifest 321 (e.g., an XML encryption manifest) that identifies portions of the body field 330 that are encrypted. Accordingly, none, some, or all portions of the body field 330 may be encrypted. The opportunity to encrypt only selected portions of the body field 330 is advantageous as this allows for the more sensitive data in the body field 330 to be encrypted without expending unneeded processor cycles encrypting the less sensitive data in the body field 330 to thereby provide a suitable balance between processing efficiency and security.

[0036] The method 300 then includes an act of including one or more security tokens in a header portion of the electronic message (act 212). The one or more security tokens may be, for example, one or more signatures. For example, the header field 310 includes a first signature 312, a possible second signature 313, and potentially other signatures 314. The first signature 312 is oval-shaped to represent that the first signature may have been signed using a corresponding credential 315, which is also represented as being oval-shaped. The second signature 313 is trapezoidal-shaped to represent that the second signature may have

been signed using a different corresponding credential 316, which is also represented as being trapezoidal-shaped.

**[0037]** In addition to including the signatures (act 212) or other security tokens in the electronic message, the method 300 includes an act of encoding one or more credentials (act 213), and then the act of including the one or more encoded credentials in the electronic message (act 214). In some cases, the credential(s) included in the electronic message may not be encoded at all thus eliminating act 213. In other cases, a credential will not be included in the electronic message either thus eliminating act 214. For example, there may not be any cause for including a credential if there is a reference to an associated credential field described below where the credential may be external to the electronic message.

**[0038]** The credentials may be, for example, any item of information that helps to identify and/or authenticate the credential provider. One type of credential is a license, which contains a set of related assertions signed by an authority. Some assertions may be about keys that may be used to sign and/or encrypt messages. Example licenses include X.509 certificates and Kerberos tickets. The owner of a license is a principle entity that can use the license authoritatively. Specifically, the principle has the knowledge necessary to apply the cryptographic keys located in the attached license or licenses attached therein. Another type of credential is biometric data about the user sending the electronic message. Such biometric data may include any information derived from the biology of the user such as retinal scan data, fingerprint data, DNA data, or any other data that may substantially uniquely identify a user based on the user's biology. In Figure 3, the included credentials are represented by first credential 315, second credential 316, and other credentials 317.

[0039] The method 200 also includes an act of including, in the header portion, an identification of an encoding format of the credential(s) (act 215). This identification is represented in Figure 3 for the first credential 315 by the encoding format field 318A. The identification is represented in Figure 3 for the second credential 316 by the encoding format field 318B. Alternatively, the encoding format is not identified in the header portion thereby indicating that a default encoding format was used to encode the credential. The method 300 also includes an act of including, in the header portion, an identification of a type of the credential (act 216). For example, the type of credential may be an X.509 certificate or a Kerberos ticket. The identification of the type of credential is represented in Figure 3 for the first credential 315 by the credential type field 319A, and for the second credential 316 by the credential type field 319B. This type of credential may be a human-readable expression or may just be any information from which the type of credential may be derived.

[0040] In this example, there is an identification of the encoding format and a credential type for each of the credentials included in the electronic message, although this is not necessary. For example, in cases in which the encoding format is the same for all of the credentials in the electronic message, the encoding format may be listed in just one portion of the electronic message. Similarly, if the credential type is the same for all of the credentials in the electronic message, the credential format may just be listed once. In addition, if a particular credential has a default encoding format (and/or credential type), then the particular encoding format (and/or credential type) need not be expressly included for that credential. Furthermore, in schema-based communication in which the schema is understood to both parties, the encoding type and/or the credential format may be implicit based on the position of the credential within the schema-based document.

[0041] Also, the identification of the encoding format and credential type are illustrated as being included in the corresponding credential field. If these fields are included in the corresponding credential field, acts 215 and 216 would occur concurrently with act 213 for that credential. However, the encoding format and type field may instead just be associated with the corresponding credential field.

[0042] The method 300 also includes an act of generating a reference indicating where a credential associated with the signature may be found (act 217), and including the reference in the header portion of the electronic message (act 218). The reference is represented in Figure 3 for the first signature 312 by the reference to associated credential field 320A, and for the second signature 313 by the reference to associated credential field 320B. While the reference may include a reference to a position internal to the electronic message (e.g., credential fields 315 and 316), the reference may also be a Uniform Resource Locator (URL) that identifies a location external to the electronic message where the associated credential may be found.

[0043] The sending computing system then transmits the electronic message to one or more recipient computing systems (act 219), which then receive the electronic message (act 220). The electronic message may contain multiple different signatures that were generated using multiple different kinds of credentials. The recipient computing system may then select one of a multiple signatures included in a header portion of the electronic message (act 221), and then read that electronic signature from the electronic message (act 222). Accordingly, the recipient computing system may choose one, some, or all of the included signatures depending on which one the recipient computing system is configured to process and trust.



[0044] The ability of the electronic message 300 to contain multiple credentials of different types allows for several novel network security configurations. For example, Figure 4A illustrates a network environment 400A in which multiple different credentials are used to identify a source computing system 401A to a particular recipient computing system 411A in a model called herein the “multiple credential – single recipient model”. In this model, a single recipient computing system 411A uses two different credentials 421 and 422 in order to authenticate the source computing system 401A. The credentials are illustrated in Figures 4A through 4D as having different shapes to emphasize that the credentials may be of different types.

[0045] Figure 4B illustrates a network environment 400B in which different credentials in the electronic message may be used to identify a source computing system 401B to an intermediary computing system 411B and to identify the source computing device 401B to a recipient computing system 412B in a model called herein the “serial credential model”. In the illustrated serial credential model 400B, the intermediary computing system 411B uses the credential 421, while the recipient computing system 412B uses the credential 422.

[0046] Figure 4C illustrates a network environment 400C in which different credentials in the electronic message may be used to identify the source computing system 401C to different recipient computing devices 411C and 412C in a model called herein the “parallel credential model”. In the illustrated parallel credential model 400C, the recipient computing system 411C uses the credential 421, while the recipient computing system 412C uses the credential 422.

[0047] There are various combinations of each of the models of Figure 4A through 4C that make a practically limitless variety of network configurations. For example, Figure 4D

illustrates a network environment 400D which combines all of the models of Figures 4A, 4B and 4C in one of many possible ways. In the environment 400D, the electronic message includes three credentials 421, 422 and 423. Credential 423 is different than credentials 421 and 422 as represented by its triangular shape. The intermediary computing system 411C uses credential 421, recipient computing system 412D uses credential 422, and recipient computing system 413D uses credentials 422 and 423.

[0048] Returning to Figure 2, the receiving computing system also may read the reference that indicates where an associated credential may be found (act 223), use that reference to find the credential (act 224), and then determine if the credential corresponds with the electronic signature (act 225). If the referenced credential corresponds to the signature, then the data signed can be associated with the credential. Accordingly, the integrity of any statements made in the credentials such as identity, rights, and so forth, may be more assured, especially if the referenced credential was external to the electronic message and thus not subject to the same tampering instances that the electronic message may be subject to.

[0049] In one embodiment, the electronic message 300 may be a Simple Object Access Protocol (SOAP) envelope although this is not required. The two provisional patent applications previously incorporated herein by reference provide several examples of SOAP envelopes which incorporate various aspects of the present invention. The following SOAP envelope is a code example of one specific embodiment of the data structure of the electronic message 300. The code example is represented in eXtensible Markup Language (XML) version 1.0. Line numbering has been added for clarity in explaining the structure of the code example. Although this code example shows one specific implementation, there

are a vast variety of different implementations that may employ the principles of the present invention. For example, although this example illustrates the use of headers hierarchically structured in a certain way and having particular header uses, other embodiments may have a different hierarchy and usage of headers without departing from the scope of the principles of the present invention.

1. `<?xml version="1.0" encoding="utf-8"?>`
2. `<S:Envelope`
3. `xmlns:S="http://schemas.xmlsoap.org/soap/envelope/"`
4. `xmlns:xsd="http://www.w3.org/2001/XMLSchema"`
5. `xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">`
6. `<Header>`
7. `<m:path xmlns:m="http://schemas.xmlsoap.org/rp">`
8. `PATH INFORMATION`
9. `</m:path>`
10. `<wssec:credentials`
11. `xmlns:wssec="http://schemas.xmlsoap.org/ws/2001/10/security">`
12. `<wslic:binaryLicence`
13. `xmlns:wslic="http://schemas.xmlsoap.org/ws/2001/10/licenses"`
14. `wslic:valueType="wslic:x509v3"`
15. `xsi:type="xsd:base64Binary"`
16. `id="X509License">`
17. `X509LICENSE ENCODED IN BASE64BINARY`

18.       </wslic:binaryLicense>

19.       <wslic:binaryCredential xmlns:tru="...">

20.               wslic:valueType="tru:binaryCredentialFormat"

21.               xsi:type="xsd:base64Binary"

22.               id="BinaryCredential">

23.                       BINARY CREDENTIAL ENCODED IN BASE64BINARY

24.       </wslic:binaryCredential>

25.       </wssec:credentials>

26.       <wssec:integrity>

27.               <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

28.                       <ds:SignedInfo>

29.                               <ds:CanonicalizationMethod

30.                                       Algorithm="http://www.w3.org/Signature/Drafts/xml-exc-c14n"/>

31.                               <ds:SignatureMethod

32.                                       Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>

33.                       <ds:Reference>

34.                               <ds:Transforms>

35.                                       <ds:Transform Algorithm="http://schemas.xmlsoap.org/

36.                                               2001/10/security#RoutingSignatureTransform"/>

37.                                       <ds:Transform Algorithm="http://www.w3.org/

38.                                               TR/2001/REC-xml-c14n-20010315/>

39.                       </ds:Transforms>

40.                       <ds:DigestMethod Algorithm="http://www.w3.org/

41.                   2000/09/xmldsig#sha1"/>  
 42.                   </ds:Reference>  
 43.                   </ds:SignedInfo>  
 44.                   <ds:SignatureValue>  
 45.                   FIRST SIGNATURE VALUE  
 46.                   </ds:SignatureValue>  
 47.                   <ds:KeyInfo>  
 48.                   <wssec:LicenseLocation="#X509License"/>  
 49.                   </ds:KeyInfo>  
 50.                   </ds:Signature>  
 51.                   <ds:Signature>  
 52.                   <ds:SignedInfo>  
 53.                   <ds:CanonicalizationMethod  
 54.                   Algorithm="http://www.w3.org/Signature/Drafts/xml-exc-c14n"/>  
 55.                   <ds:SignatureMethod  
 56.                   Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>  
 57.                   <ds:Reference>  
 58.                   <ds:Transforms>  
 59.                   <ds:Transform Algorithm="http://schemas.xmlsoap.org/  
 60.                   2001/10/security#RoutingSignatureTransform"/>  
 61.                   <ds:Transform Algorithm="http://www.w3.org/  
 62.                   TR/2001/REC-xml-c14n-20010315/>  
 63.                   </ds:Transforms>

64.                   <ds:DigestMethod Algorithm="http://www.w3.org/  
65.                   2000/09/xmlsig#sha1"/>  
66.                   </ds:Reference>  
67.                   </ds:SignedInfo>  
68.                   <ds:SignatureValue>  
69.                   SECOND SIGNATURE VALUE  
70.                   </ds:SignatureValue>  
71.                   <ds:KeyInfo>  
72.                   <wssec:LicenseLocation="#BinaryCredential"/>  
73.                   </ds:KeyInfo>  
74.                   </ds:Signature>  
75.                   </wssec:integrity>  
76.                   </S:Header>  
77.                   <S:Body>  
78.                   BODY  
79.                   </S:Body>  
80.                   </S:Envelope>

**[0050]**       Line 1 defines the XML version that the SOAP envelope follows as well as the encoding format for the SOAP envelope as a whole.

**[0051]**       Lines 2 through 80 define a SOAP envelope that includes two different credentials, two different signatures signed using the credentials, and references to the

credentials for each of the signatures. Also, the encoding type and format type of each of the credentials is specified.

[0052] Lines 3 through 5 define global namespace abbreviations used throughout the SOAP envelope. It is standard practice to specify namespace abbreviation in this portion of the SOAP envelope. These namespace abbreviations correspond to a namespace that defines a standard for how particular elements to which the namespace applies are to be interpreted.

[0053] Lines 77 though 79 represent the body of the SOAP envelope and is an example of the body field 330 of Figure 3. Note that the actual body content is replaced with the capitalized term "BODY". Capitalized terms are used throughout the code example to replace actual content whose value is not specifically included in the code example and the value is not important to the principles of the present invention. For example, the term "BODY" in line 78 could be any content without affecting the principles of the present invention.

[0054] Lines 6 though 76 represent the header information for the SOAP envelope and is an example of the header field 310 of Figure 3.

[0055] Lines 7 through 9 express the path that the electronic message is to take. Intermediary computing systems such as intermediary computing system 411B of Figure 4B may be specified in this section.

[0056] Lines 10 through 25 define a unique SOAP header called "credentials". This header may include several different credential types. The encoding format and type of the credential may also be specified in this header.

[0057] For instance, lines 12 through 18 contain a binary license (see line 17) called “X509License” (see line 16), which is identified as being an X.509 certificate (see line 14), and which is identified as being encoded using base64binary encoding (see line 15). Line 14 is an example of the credential type field 319A of Figure 3. Line 15 is an example of the encoding format field 318A of Figure 3. Line 17 is an example of the first credential field 315 of Figure 3.

[0058] Also, lines 19 through 24 contain a binary credential (see line 23) called “BinaryCredential” (see line 22), which is identified as being in a binary credential format (see line 20), and which is identified as being encoded also using base64binary encoding (see line 21). Line 20 is an example of the credential type field 319B of Figure 3. Line 21 is an example of the encoding format field 318B of Figure 3. Line 23 is an example of the second credential field 316 of Figure 3.

[0059] Lines 26 through 75 define an “integrity” header that contains two signatures, each having a reference location to find a corresponding credential that may be used to verify the integrity of the electronic message (i.e., that the electronic message was sent by the signer of the signature, and that the electronic message has not been altered in transit).

[0060] In particular, a first signature element is referenced from lines 27 through 50, with the second signature element being referenced from lines 51 through 74. Each signature element follows the schema defined by XML digital signature in accordance with the “http://www.w3.org/2000/09/xmldsig#” namespace. However, that KeyInfo child element within each XML digital signature includes a “LicenseLocation” element that references the location of a license (or other credential) that may be used to verify the integrity of the electronic message.



[0061] The first signature element includes a “SignedInfo” element from lines 28 through 43 which defines canonicalization methods, digest algorithms, and various transforms that apply to the signature. The first signature value is included at line 45 and is an example of the first signature field 312 of Figure 3. The license location specified at line 48 is an example of the reference to associated credential field 320A of Figure 3.

[0062] The second signature element is similar to the first signature element except that the second signature value is at line 69 and represents an example of second signature field 313 of Figure 3, while the license location is specified at line 72 and represents an example of the reference to associated credential field 320B of Figure 3.

[0063] Although the specific code example above includes the multiple credentials and signatures in the header portion of a SOAP envelope, the credential and/or signature information may also be included within the header portion of a HyperText Transport Protocol (HTTP) message. Accordingly, the principles of the present invention allow for the communication of one or more different credentials in a single electronic message. In addition, the reference to an associated credential allows for the integrity of electronic messages to be verified.

[0064] In the above code example, there are two different credentials included in the electronic message, a binary license and a binary credential. These credential types may be abstractly structured in an inheritance tree. A hierarchically-structured credential semantics inheritance tree that includes these credential types is illustrates as tree 500 in Figure 5.

[0065] The tree 500 includes an abstract credential data type 501 at its base. The abstract credential is structured in accordance with a schema 501A and has handling rules

501B. The schema 501A describes the basic structure of the abstract credential data type. The handling rules 501B describe how to handle the abstract credential.

[0066] One of the first-tier branches of the tree 500 is an abstract license data type 511, which includes an extended schema 511A and extended handling rules 511B. The schema and the handling rules from parent nodes in the tree 500 may be inherited by the child nodes. In other words, the schema 511A of the abstract license may reflect the schema 501A of the abstract credential with some specified extensions. Also, the handling rules 511B may represent further handling rules in addition to the handling rules 501B specified at the abstract credential data type.

[0067] Another of the first-tier branches of the tree 500 is a binary credential data type 512 that includes schema 512A and handling rules 512B. A second tier-branch of the tree includes binary license 521 having schema 521A and handling rules 521B. The tree 500 may be further expanded by defining a schema that extends on the schema of a parent node, and/or by defining further handling rules in addition to those provided for a parent node.

[0068] When determining how to structure a binary license for example, the source computing system would use the schema 521A. If the schema 521A represented incremental structural changes rather than a complete structural definition, the source computing system may also consult the schema of ancestral data types 511 and 501 to determine the final structural form of the binary license. When receiving a binary license, the recipient computing system may use the schema 521A to determine how to parse the binary license, along with the handling rules 521B to determine how to treat the binary license in terms of how to process the binary license and what authorities to grant in response to the binary license. The handling rules 521B may represent incremental handling

rules in which case the ancestral handling rules 511B and 501B may also be consulted to determine proper handling. The tree 500 may be stored at both the source computing system and the recipient computing system so as to ensure consistent treatment of credentials.

[0069] The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.